

Agentic Knowledge Graph Entity Linking

Marius Birmele^{1,*}, Sebastian Walter^{1,*}, and Hannah Bast¹
¹ University of Freiburg, Department of Computer Science * equal contribution
{birmelem,swalter}@cs.uni-freiburg.de



universität freiburg

Problem Find entity mentions in text and link them to a knowledge graph **without task-specific training**

Solution An LLM-based agent that **explores the KG** and **annotates the text** via tool calls

Entity-Linking Results

End-to-end F₁ (%) on five ELEVANT benchmarks. GRASP runs on the full Wikidata KG from Feb 2026. ReFinED (**fine-tuned on AIDA**) is ELEVANT's 2024 run on ca. 6M Wikipedia entities.

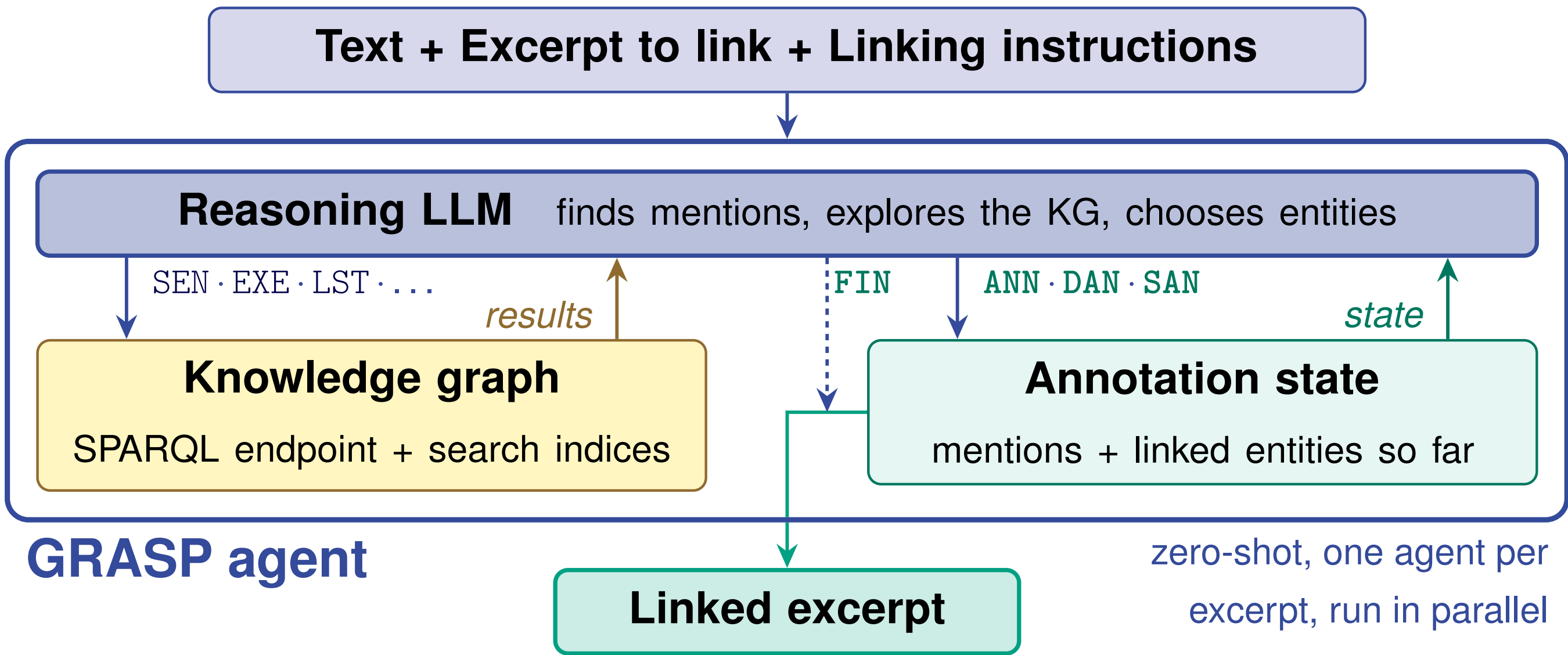
System	Wiki	News	KORE	MSNBC	AIDA	Average	
ReFinED	68.4	66.3	68.7	74.5	85.2	72.6	
GRASP (zero-shot) with							
GPT-5.4 mini	73.4	69.5	80.9	74.9	72.9	74.3	
Qwen3.6-27B	81.0	74.9	83.9	75.8	74.7	78.1	
Qwen3.8-27B-INT4	80.4	74.5	84.2	<u>77.5</u>	75.9	78.5	
Gemma 4 31B IT	80.9	76.4	83.3	<u>77.5</u>	<u>76.2</u>	78.9	
Qwen3.5-397B	81.4	75.8	85.9	77.3	74.7	79.0	
Kimi K2.7 Code	82.3	75.2	88.2	79.3	76.8	80.3	

Best per column in bold, second-best underlined. ReLiK and Mistral Small 4 are in the paper.

Targeted instructions narrow the AIDA gap. Rules for metonyms (country → national team), demonyms and capitalized words lift Qwen3.6-27B from 74.7 to **81.7**.

Approach

We extend **GRASP**, our framework for agentic KG tasks, with **annotation functions** for entity linking. The LLM finds mentions, explores the KG to disambiguate them, and annotates the text. The approach needs **no training** and is steered by natural-language **instructions**.



Knowledge-graph functions (inherited from GRASP)

SEN search entity SPR search property SPE property of entity SOP object of property
EXE execute SPARQL LST list triples SWF search with filter

Annotation functions (new for entity linking)

ANN annotate(kg, words, occ, entity) DAN delete(words, occ)
SAN show_annotations() FIN finalize()

Words + occurrence index. LLMs handle character offsets poorly, so ANN takes the words and an occurrence index. Of four methods we tried (Markdown links, word indices, prefix + suffix, occurrence index), this was the most reliable.

Feedback. ANN, DAN and SAN return the current annotation state, so the agent can check and revise it before FIN.

Grounded IDs. ANN only accepts entities that already appeared in a function call result.

GRASP vs. Claude Code

Claude Code gets the same task and Wikidata endpoint as GRASP. Wiki-Fair is sub-sampled to 20 random articles. GRASP's cost is based on API prices assuming an 80% cache hit rate.

System	KORE	MSNBC	Wiki	Average	Tokens I/O	Cost
Claude Code with						
Haiku 4.5	83.1	63.6	54.6	67.1	38.3M / 1.0M	\$12.56
Sonnet 5	86.1	79.4	77.3	80.9	45.9M / 1.3M	\$53.87
GRASP with						
Gemma 4 31B IT	83.3	77.5	<u>79.2</u>	80.0	37.0M / 0.7M	\$3.95
Kimi K2.7 Code	88.2	<u>79.3</u>	79.7	82.4	41.3M / 0.7M	\$16.92

Takeaways

GRASP **beats the fine-tuned ReFinED** on four of five benchmarks, by up to 19.5 pp.

It **outperforms** Claude Code with Sonnet 5 at **less than a third of the cost**, and is **far ahead** with smaller models (Gemma 4 31B: 80.0 vs. Haiku 4.5: 67.1).

It also runs **locally**: on a single RTX 5090, Qwen3.8-27B-INT4 averages 78.5.

Example

● **Input** Erich was born in Neunkirchen.

● **Instructions** Annotate all named entities. Do not annotate NIL.

● *A person and a place: search both, then use the birth relation.*

SEN (query="Erich")
↔ Erich (wd:Q942534) town in India, Erich (wd:Q15277196) given name, ...

● *"Erich" alone is ambiguous and NIL is not allowed: skip it for now.*

SEN (query="Neunkirchen")
↔ Neunkirchen (wd:Q6880) aka Neunkirchen (Saar), town in Saarland
Neunkirchen (wd:Q265948) town in Austria, ...

● *Take Neunkirchen (Saar). Is there a clear candidate "Erich" born there?*

EXE (sparql="SELECT ?p WHERE { ?p wdt:P19 wd:Q6880 ; rdfs:label ?l .
FILTER(STRSTARTS(LCASE(?l), \"erich\")) }")
↔ Erich Schwertner (wd:Q1353234), Erich Walch (wd:Q1353435),
Erich Honecker (wd:Q2607)

● *Three matches. When candidates compete, pick the best-fitting, more general one: the prominent Erich Honecker. Annotate both.*

ANN (words="Erich", occ=0, entity=wd:Q2607)
↔ **state**: [Erich](wd:Q2607) was born in Neunkirchen.

ANN (words="Neunkirchen", occ=0, entity=wd:Q6880)
↔ **state**: [Erich](wd:Q2607) was born in [Neunkirchen](wd:Q6880).

FIN ()

● **Output** Erich was born in Neunkirchen.

Erich Honecker

Neunkirchen (Saar)



Full trace

More examples

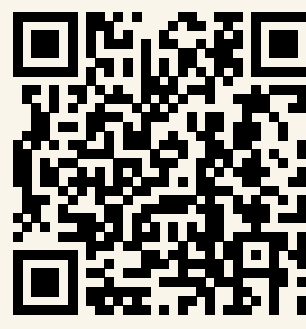
scan for the full traces

Mars, Galaxy, Bounty



Brands, not planets

Barcelona vs. Juventus



Default run



Link demonyms



Restrict types

Resources & Outlook

Paper



Read the details

Code



GRASP on GitHub

Demo



Try it yourself

Results



Inspect via ELEVANT

Outlook

Evaluate on **other knowledge graphs**

Extend to **property, class, and fact linking** and to co-reference resolution

Fine-tune with **reinforcement learning** for higher **reliability** and **efficiency**